STAT 672 Statistical Learning & Data Analytics Spring 2018

George Mason University

Instructor: Martin Slawski

Homework 2

Pencil-and-paper part due Thursday, March 1st, in class. Code and relevant output has to be submitted via Blackboard. Please follow the following format for the filename of your submission: `Lastname_Firstname_HWx.zip`, where `x` needs to be substituted by the homework #.
Maximum possible points to earn: 20; 100% = 15 points, rest counts as bonus.

Problem 1

Binary classification. Let us re-consider the following setup discussed in class.

- $\mathcal{X} = [0,1]$, $\mathcal{Y} = \{-1, 1\}$,
- The marginal distribution of X is the uniform distribution on $[0,1]$,
- $\mathbb{P}(Y = 1 | X = x) = \begin{cases} 0.9, & x < 0.2, \\ 0.2, & 0.2 \leq x \leq 0.8, \\ 0.9, & x > 0.8. \end{cases}$

3 Points:

a) Verify that the Bayes risk R^* of the classification problem is given by $R^* = 0.16$.

Consider the function classes $\mathcal{F}_d = \mathcal{P}_d$, $d \in \{0, 1, 2, \ldots\}$, where $\mathcal{P}_d$ is the space of polynomials of degree d on $\mathbb{R}$: $\mathcal{P}_d := \{f : \; x \mapsto f(x) = \sum_{k=0}^d \alpha_k x^k, \; \{\alpha_k\}_{k=0}^d \subset \mathbb{R}\}$.

6 Points:

b) Show that the excess risk (with respect to the $0-1$ loss)

$$\min_{f \in \mathcal{F}_d} R(f) - R(f^*)$$

is zero if and only if $d \geq 2$, where f^* denotes the Bayes classifier.

Hint. Consider the polynomial $\bar{f}(x) = (x - 0.2)(x - 0.8)$ and show that $\text{sign}(\bar{f}(x)) = \text{sign}(f^*(x))$ for all $x \in [0,1]$. Note that you need to consider the "if" part and the "only if" part separately.

2 Points:

c) Write code to to generate a random pair (X, Y) according to the setup given above.

Hint. First draw X, then $Y|X = x$.

We now want to learn a classifier $x \mapsto \text{sign}(f(x))$ given a sample $\mathcal{D}_n = \{(X_i, Y_i)\}_{i=1}^n$ for $f \in \mathcal{F}_d$. More specifically, given $\mathcal{D}_n$ we want to learn f via empirical risk minimization, i.e. via

$$\min_{f \in \mathcal{F}_d} R_{\text{emp}}(f).$$

STEP 1:
Since minimizing $R_{\text{emp}}(f)$ with respect to the $0-1$ loss is computationally not tractable, we choose the logistic loss instead. This yields:

$$\min_{f \in \mathcal{F}_d} \frac{1}{n} \sum_{i=1}^n \log(1 + \exp(-Y_i f(X_i))).$$

STEP 2:
Minimizing over $f \in \mathcal{F}_d$ is equivalent to minimize over the coefficients $\{\alpha_k\}_{k=0}^d$ defining the polynomial. This yields

$$\min_{\alpha_0, \ldots, \alpha_d} \frac{1}{n} \sum_{i=1}^n \log(1 + \exp\left(-Y_i(\alpha_0 + \alpha_1 X_1 + \alpha_2 X_i^2 + \ldots + \alpha_d X_i^d)\right).$$

STEP 3:
We can solve the optimization problem in the previous step by using existing implementations of logistic regression (function `glm` in `R`, function `linear_model.LogisticRegression` in `scikit-learn`, ...). The output of these implementations yields

$$\widehat{f}(x) = \widehat{\alpha}_0 + \widehat{\alpha}_1 x + \widehat{\alpha}_2 x^2 + \ldots + \widehat{\alpha}_d x^d,$$

where $(\widehat{\alpha}_0, \ldots, \widehat{\alpha}_d)$ minimize the empirical risk above. Given a datum (X, Y), our prediction is $\text{sign}(\widehat{f}(X))$ (which may be different from Y).

To make this work, we have to consider a few things:

- $\mathcal{D}_n$ has to be a converted to a feature (or design) matrix

$$\mathbf{X} = \begin{pmatrix} 1 & X_1 & X_1^2 & \ldots & X_1^d \\ \vdots & \ldots & \ldots & \ldots & \vdots \\ 1 & X_n & X_n^2 & \ldots & X_n^d \end{pmatrix}$$

- $(\mathbf{X}, \mathbf{y})$ are fed into the implementation of logistic regression, where $\mathbf{y} = (Y_1, \ldots, Y_n)^\top$.

- Depending on the implementation/software that you use, you may need to do the following:
  - convert Y to $0/1$ instead of $-1/1$.
  - remove the first column in $\mathbf{X}$ if an intercept is included by default, or set the option to fit a model without intercept.
  - In the `Python` implementation, set the parameter `C` to a large value (like `1e5`)).

4 Points:

d) Using simulated data from c), produce code realizing the above steps for, say, $d \in \{1, 2, 5\}$.

5 Points: *Advanced simulation*

e) For $d \in \{1, 2, 5\}$ and $n \in \{10, 20, 50, 100, 200, 500, 1000\}$, do the following:

1. Generate a training sample $\mathcal{D}_n$ of size n and a second independent sample $\mathcal{D}' = \{(X_i', Y_i')\}$ of size 1000.
2. Train a classifier $\widehat{f}$ according to d), using $\mathcal{D}_n$ only and compute the misclassifcation error on both $\mathcal{D}_n$ and $\mathcal{D}'$:

$$\text{err}_{\text{emp}} = \frac{1}{n}\sum_{i=1}^{n} I(Y_i \neq \text{sign}(\widehat{f}(X_i))), \qquad \text{err}_{\text{gen}} = \frac{1}{|\mathcal{D}'|}\sum_{i=1}^{|\mathcal{D}'|} I(Y_i' \neq \text{sign}(\widehat{f}(X_i'))).$$

3. Repeat 1. and 2. 50 times and compute the average over the above two errors.

Plot the averages of err_{emp} and err_{emp} in dependency of n in the same plot, separately for each d, and try to interpret the results.